**Эзотерический язык программирования** — [язык программирования](https://ru.wikipedia.org/wiki/%D0%AF%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F), разработанный для исследования границ возможностей разработки языков программирования, для доказательства потенциально возможной реализации некой идеи, в качестве [произведения программного искусства](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B5_%D0%B8%D1%81%D0%BA%D1%83%D1%81%D1%81%D1%82%D0%B2%D0%BE) или в качестве шутки.

Многие эзотерические языки придумываются для [развлечения](https://ru.wikipedia.org/wiki/%D0%97%D0%B0%D0%BD%D0%B8%D0%BC%D0%B0%D1%82%D0%B5%D0%BB%D1%8C%D0%BD%D0%B0%D1%8F_%D0%BC%D0%B0%D1%82%D0%B5%D0%BC%D0%B0%D1%82%D0%B8%D0%BA%D0%B0), часто они пародируют «настоящие» или являются абсурдным воплощением «серьёзных» концепций программирования. Общее свойство, присущее любому эзотерическому языку — текст программы на нём понятен лишь «посвящённому» либо непонятен вообще, потому что для составления программы нужно написать программу на обычном языке программирования. В то время как разработчики «реальных» языков программирования стараются сделать синтаксис максимально понятным, а программирование — удобным, создатели эзотерических языков ставят перед собой другие задачи.

По [словам](https://tomassetti.me/discovering-arcane-world-esoteric-programming-languages/) архитектора программного обеспечения Федерико Томассетти (Federico Tomassetti), эзотерический язык программирования разрабатывается с целью бросить вызов «нормам» проектирования ЯП. Вызов этот может выражаться по-разному: от простого стремления к чему-то необычному до создания полноценного «художественного произведения».

Важным моментом является то, что авторы многих эзотерических языков стремятся заложить в их синтаксис и структуру как можно больше отличий (от всего, что было создано в этой сфере ранее). Поэтому любая классификация эзотерических ЯП — это, скорее, возможность разобраться в том, как, почему и с какими целями создавались языки, нежели попытка создать строгую и упорядоченную систему.

Какие бывают варианты классификации

Полнота по Тьюрингу

Наиболее очевидный подход к разделению эзотерических языков на группы — в соответствии с тем, являются ли они **тьюринг-полными**: можно ли (хотя бы в теории) с помощью языка реализовать любую вычислимую функцию или нет. Одним из Тьюринг-полных языков является, например [INTERCAL](https://esolangs.org/wiki/INTERCAL_Turing-completeness_proof). На [сайте](http://www.catb.org/~esr/intercal/) американского программиста, хакера и сооснователя Open Source Initiative [Эрика Рэймонда](https://en.wikipedia.org/wiki/Eric_S._Raymond) (Eric S. Raymond, ESR), INTERCAL описан как «разработанный с целью добиться полноты по Тьюрингу и максимальной непохожести на существующие языки программирования».

Благодаря этим особенностям INTERCAL, по словам ESR, способен «вызвать слезы (смеха) у сильных мужчин» и является «центром международного сообщества техномазохистов».

Также следует отметить использование модификатора PLEASE: в этой программе он должен встретиться 4 или 5 раз, в произвольных местах. 3 и меньше вызовут ошибку “ICL079I PROGRAMMER IS INSUFFICIENTLY POLITE”, 6 и больше — “ICL099I PROGRAMMER IS OVERLY POLITE”.

Остальные команды и выражения сравнительно обычны: # — префикс константы, <- — присвоение, SUB — индекс массива. Первая строка примера создает массив 16-битных целых из 13 элементов.

С другой стороны, среди эзотерических языков (в силу их особенностей) есть и **неполные по Тьюрингу языки**. Такие, например, как HQ9+, имеющий всего 4 команды (из них как раз и составлено название языка). Этот шуточный язык позволяет без труда решить все «стандартные» задачи начинающего программиста: от «Hello, world!» (команда H) до вывода квайна (команда Q).

Однако даже полнота по Тьюрингу не означает, что эзотерический язык можно с легкостью использовать в работе. Среди эзотерических языков довольно часто встречаются «[тьюринговские трясины](https://en.wikipedia.org/wiki/Turing_tarpit)» — языки, обладающие полнотой по Тьюрингу, но при этом имеющие ограниченный (и даже бедный) синтаксис и семантику.

В теории они имеют те же возможности, что и любые тьюринг-полные языки (как эзотерические, так и рабочие), однако в связи с ограничениями реализовать некоторые виды программ с их помощью крайне сложно. INTERCAL, кстати, часто относят к «трясинам». Еще один пример — другой не менее популярный эзотерический ЯП Brainfuck. Чтобы представить, насколько сложны могут быть достаточно простые операции с использованием таких языков, вот пример классического «Hello, world!» на Brainfuck.

Команды брейнфак слайд

Цели создания языка

Федерико Томассетти [выделяет](https://tomassetti.me/discovering-arcane-world-esoteric-programming-languages/) несколько «целевых групп» эзотерических языков. Среди них, например:

*1. Языки, призванные решить конкретную задачу.* Не всегда эта задача является общепризнанно важной, однако для создателя языка и сообщества его единомышленников она имеет ценность — а значит, ценен и язык, который помогает ее решить. К таким можно отнести [Thue](https://en.wikipedia.org/wiki/Thue_(programming_language)) (назван в честь норвежского математика [Акселя Туэ](https://en.wikipedia.org/wiki/Axel_Thue)) — язык, созданный как демонстрация нулевого типа в иерархии Хомского (кстати, он тоже относится к «тьюринговским трясинам»).

*2. «Запутывающие языки» или языки-черные ящики*

Они призваны сделать код максимально нечитаемым для непосвященных и затруднить написание кода. Наиболее яркая особенность «черных ящиков» — обфускация, то есть намеренное сокрытие истинного значения того или иного участка кода. К таким, например, относится Malbolge (назван в честь восьмого круга ада Данте, куда попадают, в том числе, разного рода обманщики). Свидетельством его сложности может служить хотя бы то, что работающий [квайн](http://www.matthias-ernst.eu/malbolge/old_quine.mb) на Malbolge был написан через 14 лет после создания языка, [вторая версия](http://www.matthias-ernst.eu/malbolgequine.html) — еще через 3 года.

Синтаксин круга ада

*3. Языки, созданные для подтверждения жизнеспособности определенной концепции*

Сами по себе эти языки могут быть неудобными и нефункциональными, но тем не менее, они — неопровержимое свидетельство того, что концепция имеет право на существование. К таким языкам можно отнести, например, семейство языков [Funges](https://en.wikipedia.org/wiki/Esoteric_programming_language#Funges), использующее многомерные представления программ.

Шуточные языки

Иногда «языки-шутки» выделяют в отдельную группу. Вики-портал esolang, посвященный эзотерическим языкам, [описывает](http://esolangs.org/wiki/Joke_language_list) шуточные языки как «не представляющие интереса — за исключением их юмористической составляющей», а также «совершенно неподходящие для программирования даже в теории, банальные и менее интересные версии существующих эзотерических языков». Тем не менее, один только список шуточных языков на esolang насчитывает более 130 наименований. С другой стороны, к шуточным языкам иногда относят и более сложные, признанные эзотерические ЯП. Например INTERCAL, весь мануал по которому представляет шутку: во введении, в частности, [подчеркивается](http://www.muppetlabs.com/~breadbox/intercal-man/s01.html), что «все совпадения языков программирования, описанных здесь, с другими языками программирования, живыми или мертвыми, случайны».

Как правило, каждый эзотерический язык можно классифицировать сразу несколькими способами — по тьюринг-полноте, целям и задачам (если таковые четко обозначены), по привязке к художественным произведениям и даже по чувству юмора создателей языка. В конце концов, эзотерические языки для того и создаются, чтобы нарушать все правила — в том числе и правила классификации.